Instituto Federal do Sul de Minas Gerais

Projeto e Análise de Algoritmos

Aula 06 – Notação O

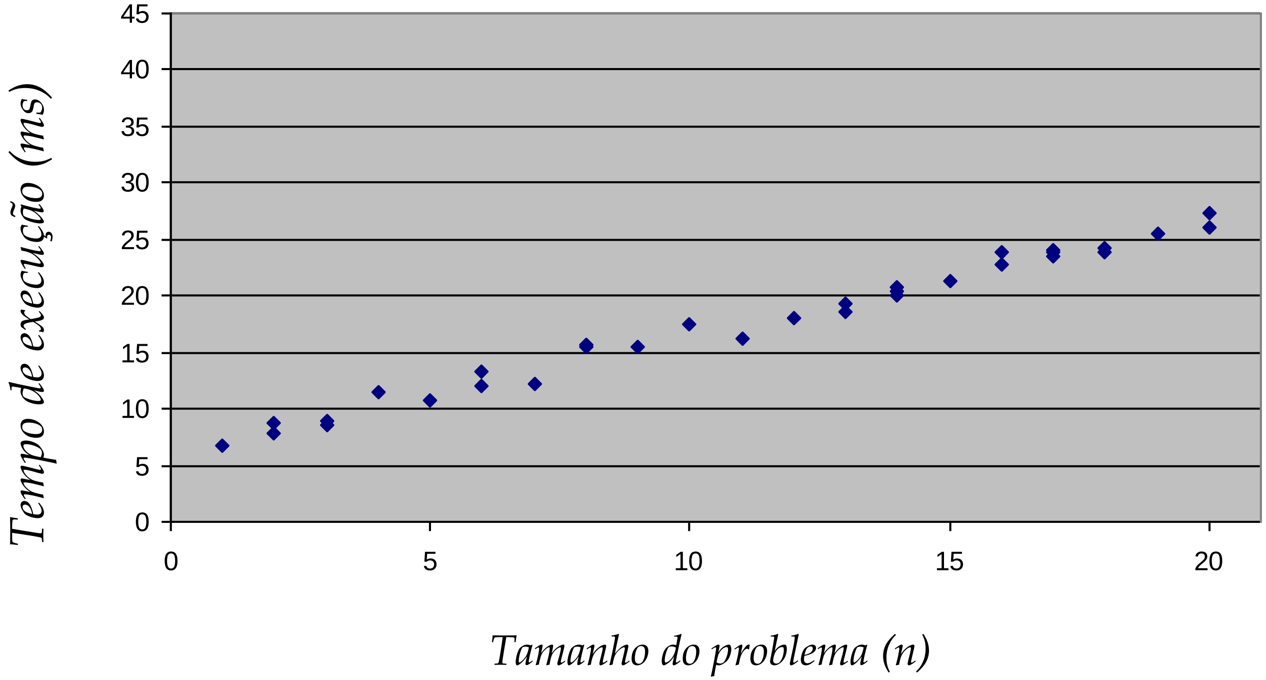
humberto@bcc.unifal-mg.edu.br

douglas@bcc.unifal-mg.edu.br

INSTITUTO FEDERAL DE
EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
SUL DE MINAS GERAIS

Última aula teórica

- Análise experimental



Análise de algoritmos

- Existem vários componentes que precisamos definir antes de descrever uma metodologia de análise de algoritmos baseada em funções matemáticas:

Análise de algoritmos

- Existem vários componentes que precisamos definir antes de descrever uma metodologia de análise de algoritmos baseada em funções matemáticas:
  - Uma linguagem para descrição de algoritmos;

Análise de algoritmos

- Existem vários componentes que precisamos definir antes de descrever uma metodologia de análise de algoritmos baseada em funções matemáticas:
  - Uma **linguagem** para descrição de algoritmos;
  - Um modelo computacional para execução de algoritmos;

Análise de algoritmos

- Existem vários componentes que precisamos definir antes de descrever uma metodologia de análise de algoritmos baseada em funções matemáticas:
  - Uma **linguagem** para descrição de algoritmos;
  - Um **modelo computacional** para execução de algoritmos;
  - Uma métrica para medir o tempo de execução de algoritmos;

Pseudo-código

Uma linguagem para descrição de algoritmos

Pseudocódigo

- Pseudocódigo é uma mistura de linguagem natural e estruturas de programação de alto nível;

Pseudocódigo

- **Pseudocódigo** é uma mistura de **linguagem natural** e **estruturas de programação** de **alto nível**;

- É utilizado para **descrever** algoritmos de **forma genérica**;

Pseudocódigo

- **Pseudocódigo** é uma mistura de **linguagem natural** e **estruturas de programação** de **alto nível**;
- É utilizado para **descrever** algoritmos de **forma genérica**;
- No entanto, **não existe uma definição precisa** da linguagem pseudo-código por causa de seu uso da linguagem natural;

Pseudocódigo

- **Pseudocódigo** é uma mistura de **linguagem natural** e **estruturas de programação** de **alto nível**;
- É utilizado para **descrever** algoritmos de **forma genérica**;
- No entanto, **não existe uma definição precisa** da linguagem pseudo-código por causa de seu uso da linguagem natural;
- Para aumentar sua clareza, o pseudocódigo mistura construções formais;

Pseudocódigo

- **Pseudocódigo** é uma mistura de **linguagem natural** e **estruturas de programação** de **alto nível**;
- É utilizado para **descrever** algoritmos de **forma genérica**;
- No entanto, **não existe uma definição precisa** da linguagem pseudo-código por causa de seu uso da linguagem natural;
- **Para aumentar sua clareza, o pseudocódigo mistura construções formais;**
- Na análise de algoritmos, devemos tomar muito cuidado com as chamadas informais.

Pseudocódigo

```
Algoritmo recebe (n)
Entrada: n é um número inteiro.
Declare:
   x, y, raiz: Inteiros;
x ← y ← 0; //Inicializando os pontos no centro
raiz = arredondar( sqrt ( n ) );
Se raiz é par então
   Se n > raiz² + raiz então
      y ← - raiz/2;
      x ← (x - raiz/2) +(n – raiz²);
   Senão
      y ← y - raiz/2 + ( n - (raiz² + raiz) );
      x ← x + raiz/2;
   Fim se
Senão //raiz é impar
   Se n > raiz² + raiz então
      y ← y + raiz/2 + 1;
      x ← (x + raiz/2) - (n – raiz²);
   Senão
      y ← y + raiz/2 + 1 - ( n - (raiz² + raiz) );
      x ← x - raiz/2 - 1;
   Fim se
Fim se
Retorne x, y;
```

O pseudocódigo sempre deve considerar:

- Expressões;
- Declarações de variáveis;
- Declarações de métodos;
- Estruturas de decisão;
- Estruturas de Repetição;
- Indexação de arranjos;
- Chamadas de métodos;
- Retorno de variáveis;

RAM

Um modelo computacional para execução de algoritmos

Modelo da máquina de acesso aleatório (RAM)

- RAM define uma máquina abstrata.

Modelo da máquina de acesso aleatório (RAM)

- **RAM** define uma **máquina abstrata**.
- Este modelo não deve ser confundido com “memória de acesso a aleatório”;

Modelo da máquina de acesso aleatório (RAM)

- **RAM** define uma **máquina abstrata**.
- Este modelo não deve ser confundido com “memória de acesso a aleatório”;
- Este modelo vê o computador com uma Unidade Central de Processamento conectada a uma memória;

Modelo da máquina de acesso aleatório (RAM)

- **RAM** define uma **máquina abstrata**.
- Este modelo não deve ser confundido com “memória de acesso a aleatório”;
- Este modelo vê o computador com uma Unidade Central de Processamento conectada a uma memória;
- Cada posição da memória armazena uma palavra, que pode ser:
  - Um número;
  - Uma cadeia de caracteres;
  - Ou um endereço.

Modelo da máquina de acesso aleatório (RAM)

- **RAM** define uma **máquina abstrata**.
- Este modelo não deve ser confundido com “memória de acesso a aleatório”;
- Este modelo vê o computador com uma Unidade Central de Processamento conectada a uma memória;
- Cada posição da **memória armazena uma palavra**, que pode ser:
  - Um **número**;
  - Uma **cadeia de caracteres**;
  - Ou um **endereço**.
- Ou seja, algum tipo básico da linguagem;

Modelo da máquina de acesso aleatório (RAM)

- O termo "acesso aleatório" refere-se à capacidade da CPU acessar uma posição arbitrária de memória em apenas uma operação primitiva;

Modelo da máquina de acesso aleatório (RAM)

- O termo "acesso aleatório" refere-se à capacidade da CPU acessar uma posição arbitrária de memória em apenas uma operação primitiva;

- Para manter o modelo simples, vamos considerar que este não possui limitação com relação ao número de itens que podem ser armazenados;
  - Assim como a Máquina de Turing!

Modelo da máquina de acesso aleatório (RAM)

- O termo "acesso aleatório" refere-se à capacidade da CPU acessar uma posição arbitrária de memória em apenas uma operação primitiva;

- Para manter o **modelo simples**, vamos considerar que **este não possui limitação com relação ao número de itens** que podem ser armazenados;
  - Assim como a Máquina de Turing!

- Presumimos também que a CPU do modelo RAM pode realizar qualquer operação primitiva em um número constante de passos que não depende do tamanho da entrada;

Modelo da máquina de acesso aleatório (RAM)

- Definição mais completa de RAM:
  - http://en.wikipedia.org/wiki/Random_access_machine

- Para os curiosos:
  - Definição de PRAM:
    - http://en.wikipedia.org/wiki/Parallel_Random_Access_Machine

Contagem de Operações Primitivas

Contagem de Operações Primitivas

- Faremos agora, depois de definido o modelo computacional, a contagem de operações primitivas de algoritmos.

```
Algoritmo arrayMax(A, n)
    entrada : um arranjo A com n ≥ 1 elementos inteiros
    Saída : o maior elemento do Arranjo A
Início
    resultado ← A[0]
    para i ← 1 até n - 1 faça
        se resultado < A[i] então
            resultado ← A[i]
        fim se
    fim para
    retorne resultado
fim
```

Contagem de Operações Primitivas

Algoritmo arrayMax(A, n)

    entrada : um arranjo A com $n \geq 1$ elementos inteiros

    Saída : o maior elemento do Arranjo A

Início

    resultado $\leftarrow$ A[0]

    para i $\leftarrow$ 1 até n - 1 faça

        se resultado < A[i] então

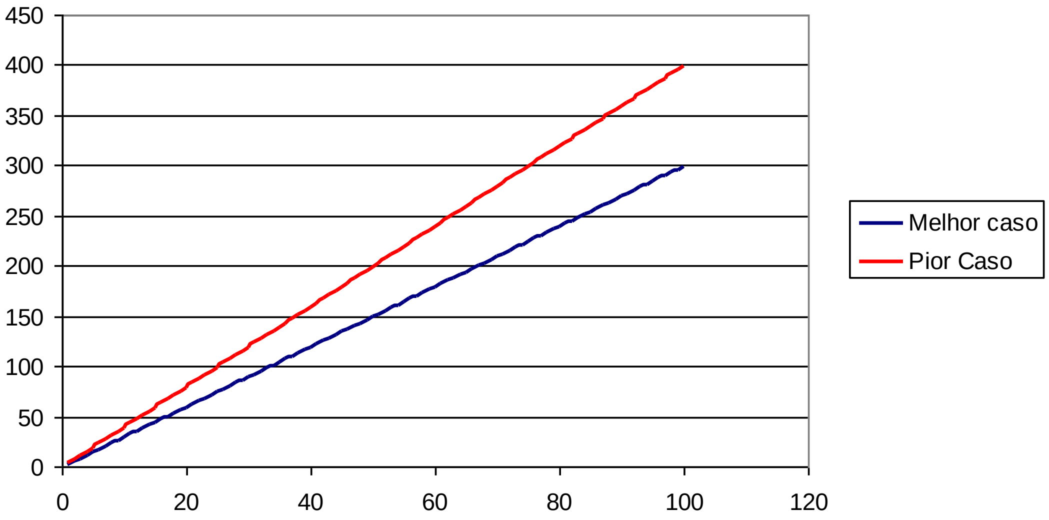
            resultado $\leftarrow$ A[i]

        fim se

    fim para

    retorne resultado

fim



Notação assintótica

Notação assintótica

- Quando observamos tamanhos de entrada grandes o suficiente para tornar relevante apenas a ordem de crescimento do tempo de execução, estamos estudando a eficiência assintótica dos algoritmos;

Notação assintótica

- **Quando observamos tamanhos de entrada grandes o suficiente** para tornar relevante apenas a ordem de crescimento do tempo de execução, **estamos estudando a eficiência assintótica dos algoritmos**;

- Ou seja, estamos preocupados com a maneira como o tempo de execução de um algoritmo aumenta, à medida que o tamanho da entrada da entrada aumenta INDEFINIDAMENTE;

Notação assintótica

- **Quando observamos tamanhos de entrada grandes o suficiente** para tornar relevante apenas a ordem de crescimento do tempo de execução, **estamos estudando a eficiência assintótica dos algoritmos**;
- Ou seja, **estamos preocupados com a maneira como o tempo de execução de um algoritmo aumenta**, à medida que o tamanho da entrada da **entrada aumenta INDEFINIDAMENTE**;
- Em geral, um algoritmo que é assintoticamente mais eficiente será a melhor escolha para todas as entradas, exceto as muito pequenas.

Notação assintótica

- A análise de um algoritmo geralmente considera com apenas algumas operações elementares.
  - Comparações;
  - Atribuições;

Notação assintótica

- A análise de um algoritmo geralmente **considera com apenas algumas operações elementares**.
  - Comparações;
  - Atribuições;

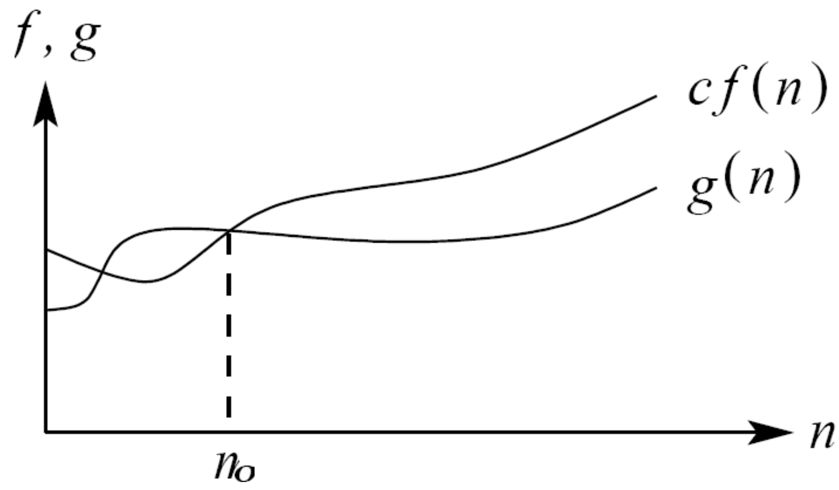
- Este fator **varia de autor para autor** na literatura;

  - Na prática, **as comparações são mais demoradas que atribuições**. Sendo assim, alguns analistas consideraram apenas comparações na análise de algoritmos.

Notação assintótica

- A medida de custo ou medida de complexidade relata o crescimento assintótico da operação considerada.

- Definição: Uma função $f(n)$ domina assintoticamente outra função $g(n)$ se existem duas constantes positivas
  - c e n_o
- tais que, para qualquer
  - $n >= n_o$,
- temos
  - $g(n) <= c \cdot f(n)$



Notação assintótica

- Escrevemos $g(n) = O(f(n))$ ou $g(n) \in O(f(n))$ para expressar que $f(n)$ domina assintoticamente $g(n)$. Encontramos leituras nos modos:
  - $g(n)$ é da ordem no máximo $f(n)$; *// formal*
  - $g(n)$ é O de $f(n)$; *// informal*
  - $g(n)$ é igual a O de $f(n)$; *// informal*
  - $g(n)$ pertence a O de $f(n)$; *// formal*

Notação assintótica

- Escrevemos $g(n) = O(f(n))$ ou $g(n) \in O(f(n))$ para expressar que $f(n)$ domina assintoticamente $g(n)$. Encontramos leituras nos modos:
  - $g(n)$ é da ordem no máximo $f(n)$; // **formal**
  - $g(n)$ é O de $f(n)$; // **informal**
  - $g(n)$ é igual a O de $f(n)$; // **informal**
  - $g(n)$ pertence a O de $f(n)$; // **formal**

- Exemplo: quando dizemos que o tempo de execução $T(n)$ de um programa é $O(n^2)$, significa que existem constantes
  - c e n_o
- tais que, para valores de
  - $n >= n_o$,
- temos:
  - $T(n) <= c.n^2$

Notação assintótica

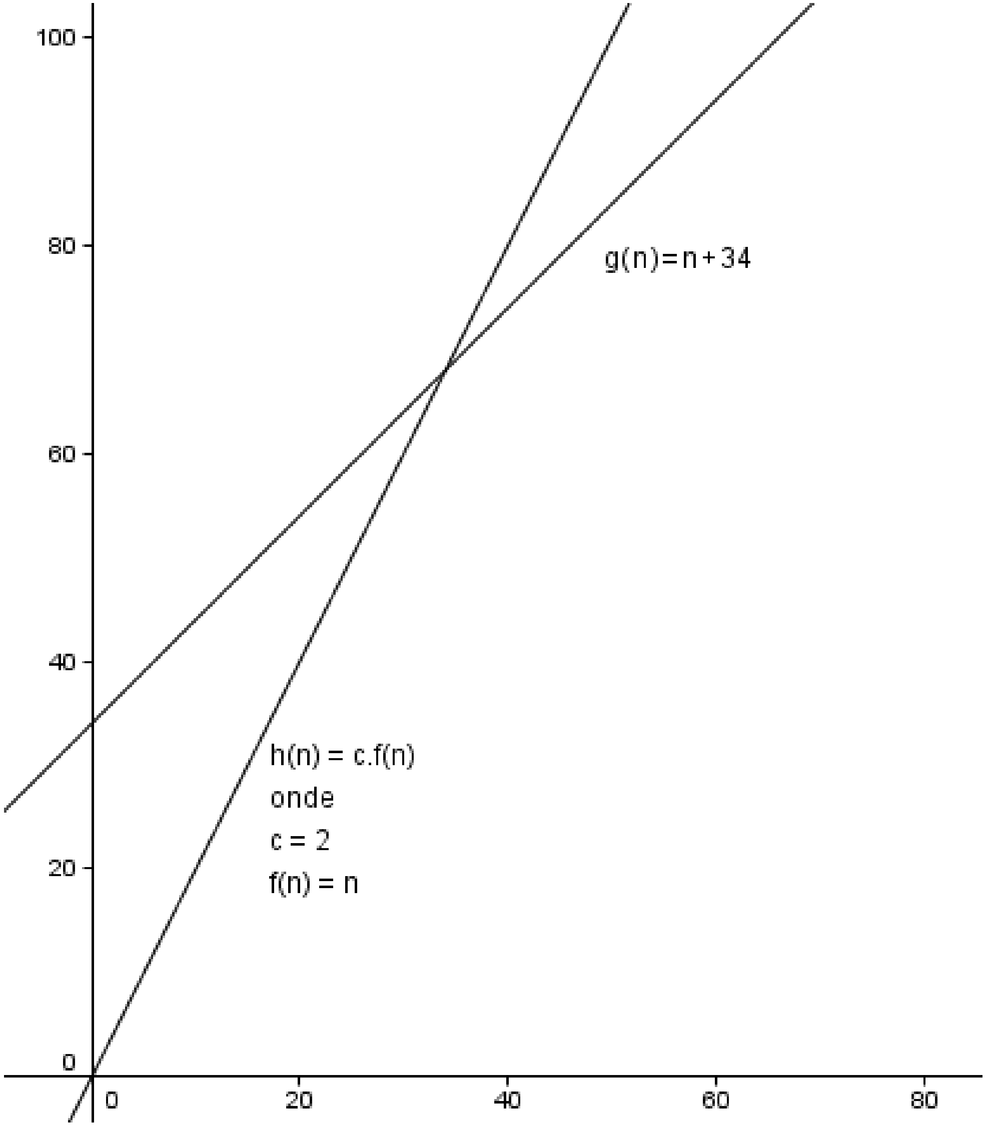
- Exemplo:
  - $f(n) = n$
  - $g(n) = n+34$

  - $g(n) \in O(f(n))???$
  - *ou*
  - $n+34 \in O(n)???$

Notação assintótica

- Exemplo:
  - $f(n) = n$
  - $g(n) = n+34$

  - $g(n) \in O(f(n))$???
  - ou
  - $n+34 \in O(n)$???



Notação assintótica

- Uma visão um pouco diferente do que a já vista por vocês em Estrutura de Dados...

- Questão 1:
  - n é $O(n^2)$?

Notação assintótica

- Questão 1:
  - n é $O(n^2)$?
  - Sim.
  - Para $n >= 1$,
    - $n <= n^2$.

Notação assintótica

- Questão 2:
  - n^2 é $O(n)$?

Notação assintótica

- Questão 2:
  - $n^2 \in O(n)$?
  - Não.
  - Suponha:

$$\exists c, n_0$$

$$\forall n \geq n_0$$

$$n^2 \leq c \cdot n$$

Notação assintótica

- Questão 2:
  - $n^2 \in O(n)$?
  - Não.
  - Suponha:

$$\exists c, n_0$$

$$\forall n \geq n_0$$

$$n^2 \leq c \cdot n$$

Absurdo, pois
c é uma constante e
n assume valores até o infinito

$$\boxed{\begin{array}{l} n^{\cancel{2}} \leq c \cdot \cancel{n} \\ n \leq c \end{array}}$$

Portanto, $\nexists c, n_0$

Notação assintótica

- Operações básicas com a notação O

$$
\begin{aligned}
f(n) &= O(f(n)) \\
c \times O(f(n)) &= O(f(n)) \qquad c = constante \\
O(f(n)) + O(f(n)) &= O(f(n)) \\
O(O(f(n))) &= O(f(n)) \\
O(f(n)) + O(g(n)) &= O(max(f(n), g(n))) \\
O(f(n))O(g(n)) &= O(f(n)g(n)) \\
f(n)O(g(n)) &= O(f(n)g(n))
\end{aligned}
$$

Exercícios

(a) $10^{56} \cdot n^2 \in O(n^2)$?

(b) $10^{56} \cdot n^2 \in O(n^3)$?

(c) $10^{56} \cdot n^2 \in O(n)$?

(d) $2^{n+1} \in O(2^n)$?

(e) $2^{2n} \in O(2^n)$?

(f) $n \in O(n^3)$?

Próxima aula

Leitura para próxima aula

- Algoritmos – Cormen
  - 3 Crescimento de funções
    - 3.1 Notação assintótica
      - Notação O
      - Notação Ω
      - Notação o
      - Notação ω
      - Comparação de funções (inclui notação θ)

Reforçando…

A complexidade de um algoritmo não está diretamente ligada o tamanho do conjunto de regras do mesmo.

Bibliografia

- CORMEN, T. H.; LEISERSON, C. E.; RIVEST, R. L.; (2002). Algoritmos – Teoria e Prática. Tradução da 2ª edição americana. Rio de Janeiro. Editora Campus.

- TAMASSIA, ROBERTO; GOODRICH, MICHAEL T. (2004). Projeto de Algoritmos - Fundamentos, Análise e Exemplos da Internet.